1. What is the smallest bunch satisfying the given equation. Express your answer formally. No proof needed.

(a) [3] \( B = 0, 2 \times B + 1 \)

(b) [3] \( B = 2, B \times B \)

2. [12] Let \( i \) be an extended integer variable, and let \( t \) be time, in the refinement

\[
P \iff \begin{cases} 
    \text{if } i = 0 \text{ then } \text{ok} \text{ else } i := i-1. \quad t := t+1. \end{cases} \quad P \fi
\]

Using recursive construction starting with \text{ok}, find a solution for \( P \). You do not need to verify that it is a solution.

3. [6] The notation \( \text{do } P \text{ while } b \text{ od} \) has been used as a loop construct that is executed as follows. First \( P \) is executed; then \( b \) is evaluated, and if \( b \) is \( \top \), execution is repeated, and if \( b \) is \( \bot \), execution is finished. Define \( \text{do } P \text{ while } b \text{ od} \) by ordinary construction and induction axioms. You can ignore time.

4. Let \( a \), \( c \), and \( x \) be natural variables. Variables \( a \) and \( c \) are implementer's variables, and \( x \) is a user's variable for the operations

\[
\begin{align*}
\text{start} & \equiv a := 1. \quad c := 0 \\
\text{double} & \equiv a := a \times 2. \quad c := c + 1 \\
\text{ask} & \equiv x := c
\end{align*}
\]

Operation \( \text{start} \) starts variable \( a \) at \( 1 \). Then repeated use of operation \( \text{double} \) doubles it some number of times. Variable \( c \) counts how many times \( a \) is doubled. Operation \( \text{ask} \) asks how many times \( a \) has been doubled since the last \( \text{start} \) operation. Reimplement this theory replacing the old implementer's variable \( a \) with nothing.

(a) [6] What is the data transformer? Prove it is a data transformer.

(b) [12] Using your data transformer, transform \( \text{double} \).